

66760" 66764" 66768

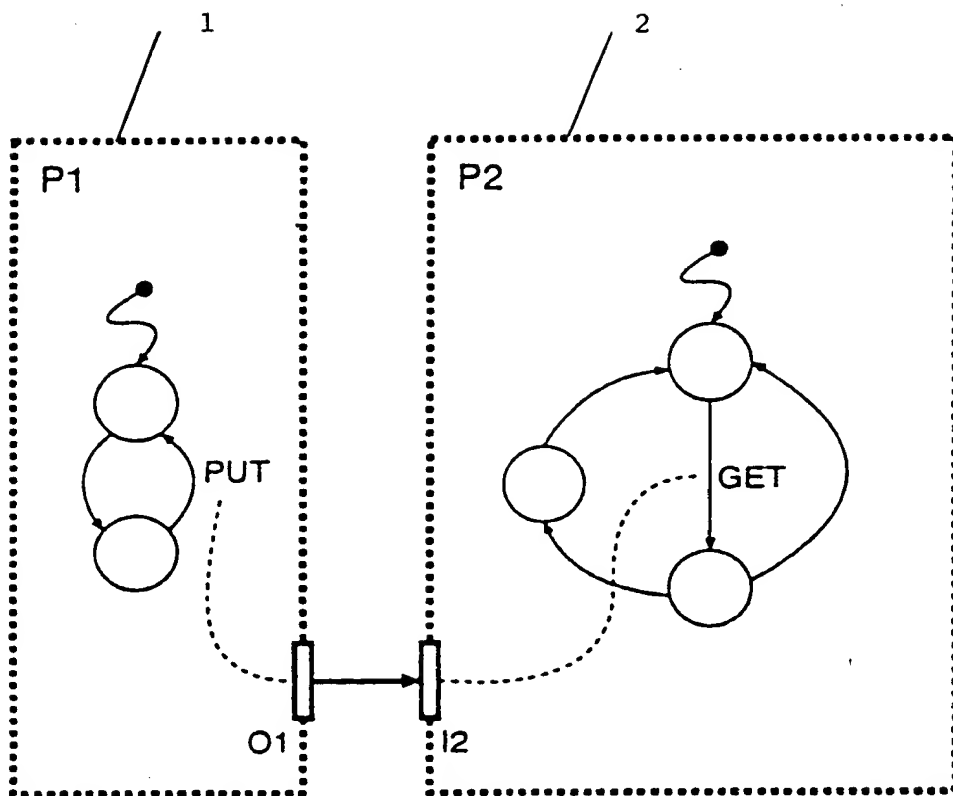


Figure 1

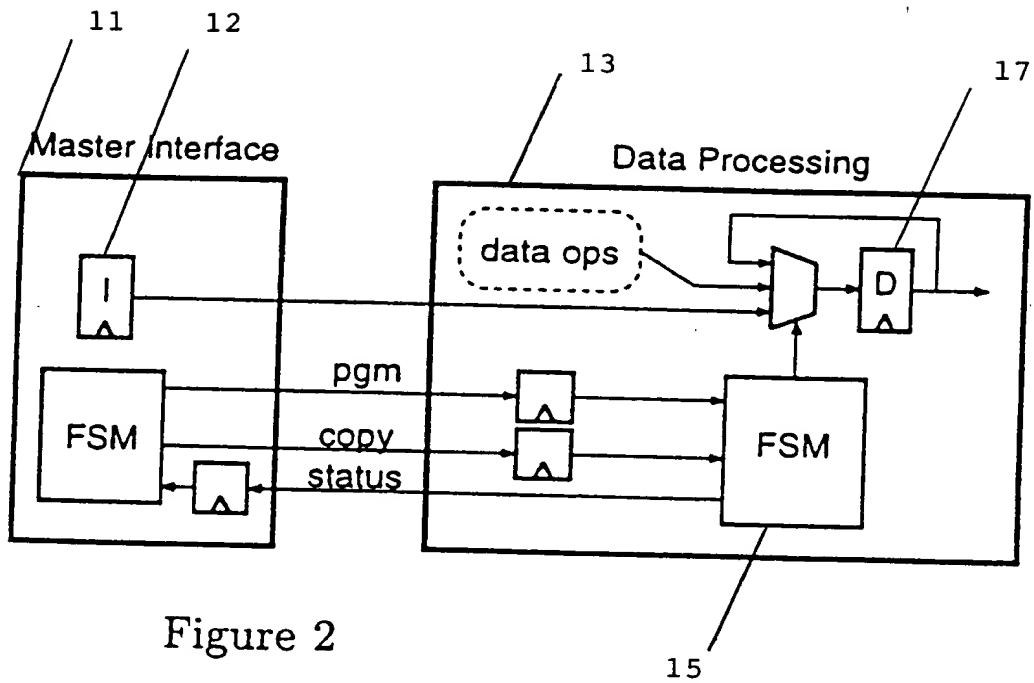


Figure 2

669722" 68964550

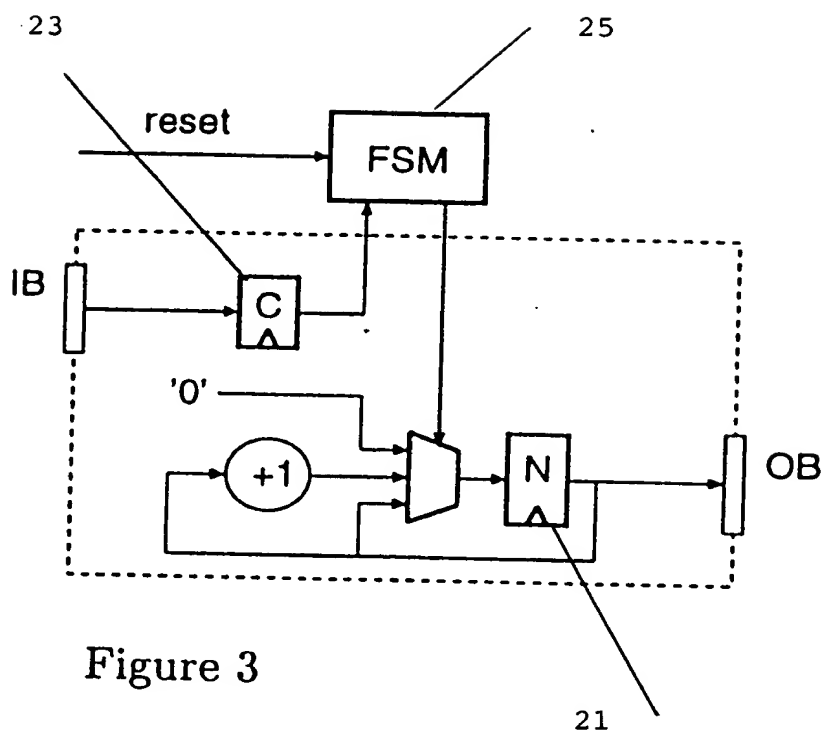
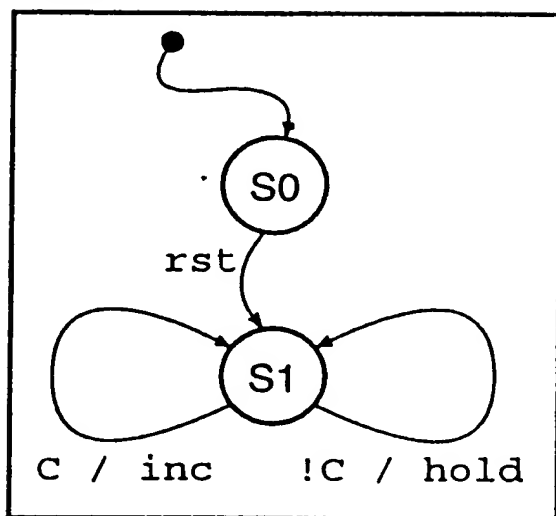


Figure 3

66250-66250

FSM



```
register C;  
register N;  
input    IB;  
output   OB;
```

```
rst:    N = 0;  
        C = IB;
```

```
inc:    N = N + 1;  
        C = IB;  
        OB = N;
```

```
hold:   C = IB;  
        OB = N;
```

Figure 4

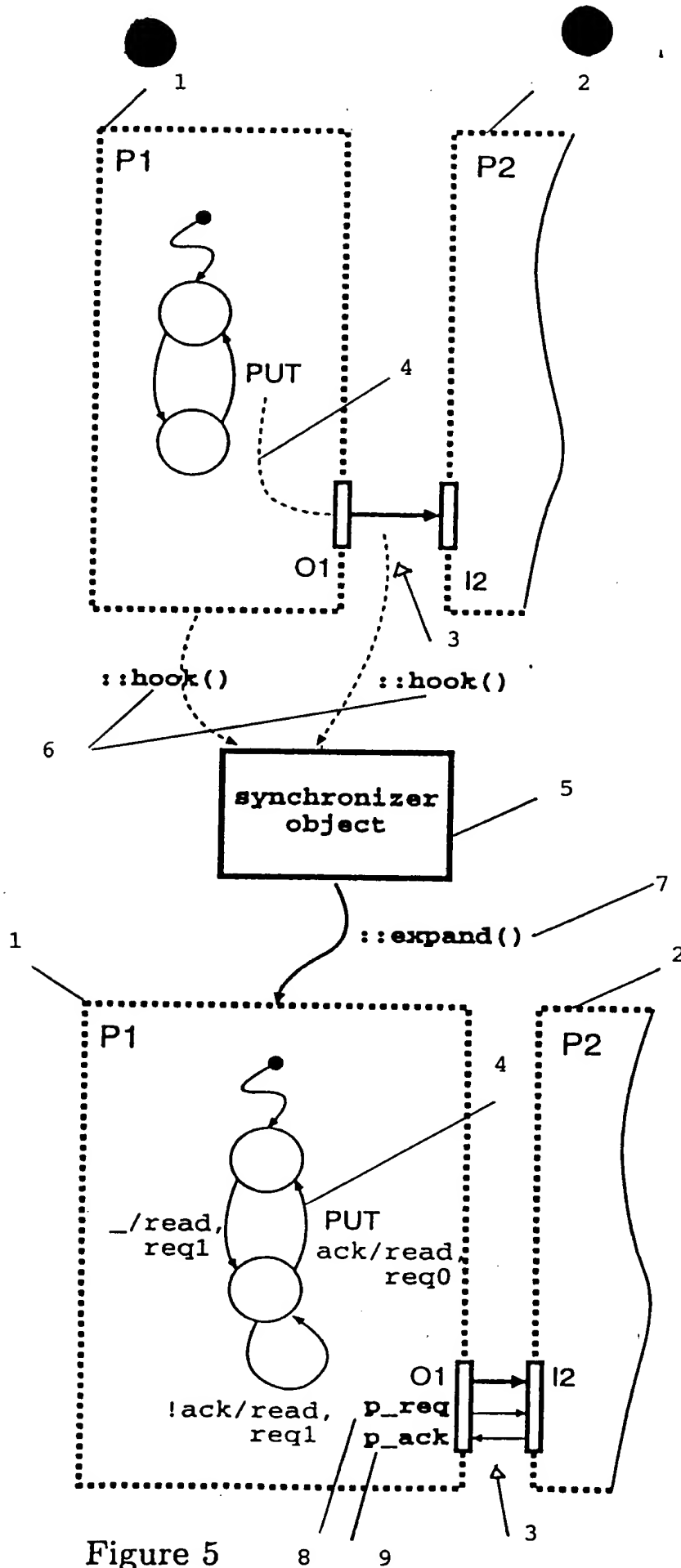


Figure 5



The diagram illustrates the state transition logic for the program iterator. It consists of two main parts: a state machine and a control logic block.

**State Machine:**

- States:** S1, S2, and a start state (represented by a circle with an incoming arrow from the top).
- Transitions:**
  - From start state to S1: `prog & !copy / read, status1`
  - From S1 to S2: `prog & !copy / read, status1`
  - From S2 to start state: `prog & copy / read, status1, upd`
  - From start state to start state (self-loop): `prog & !copy / read, status1`
  - From S2 to S2 (self-loop): `prog & copy / read, status1, upd`

**Control Logic Block:**

- Components:** A box labeled `prog_itf object` (35) and a box labeled `D` (41).
- Interactions:**
  - A dashed arrow labeled `::hook()` (45) points from the `prog_itf object` to the state machine.
  - A dashed arrow labeled `::hook()` points from the `D` box to the `prog_itf object`.
  - A solid arrow labeled `::expand()` (47) points from the `prog_itf object` to the state machine.

Figure 7

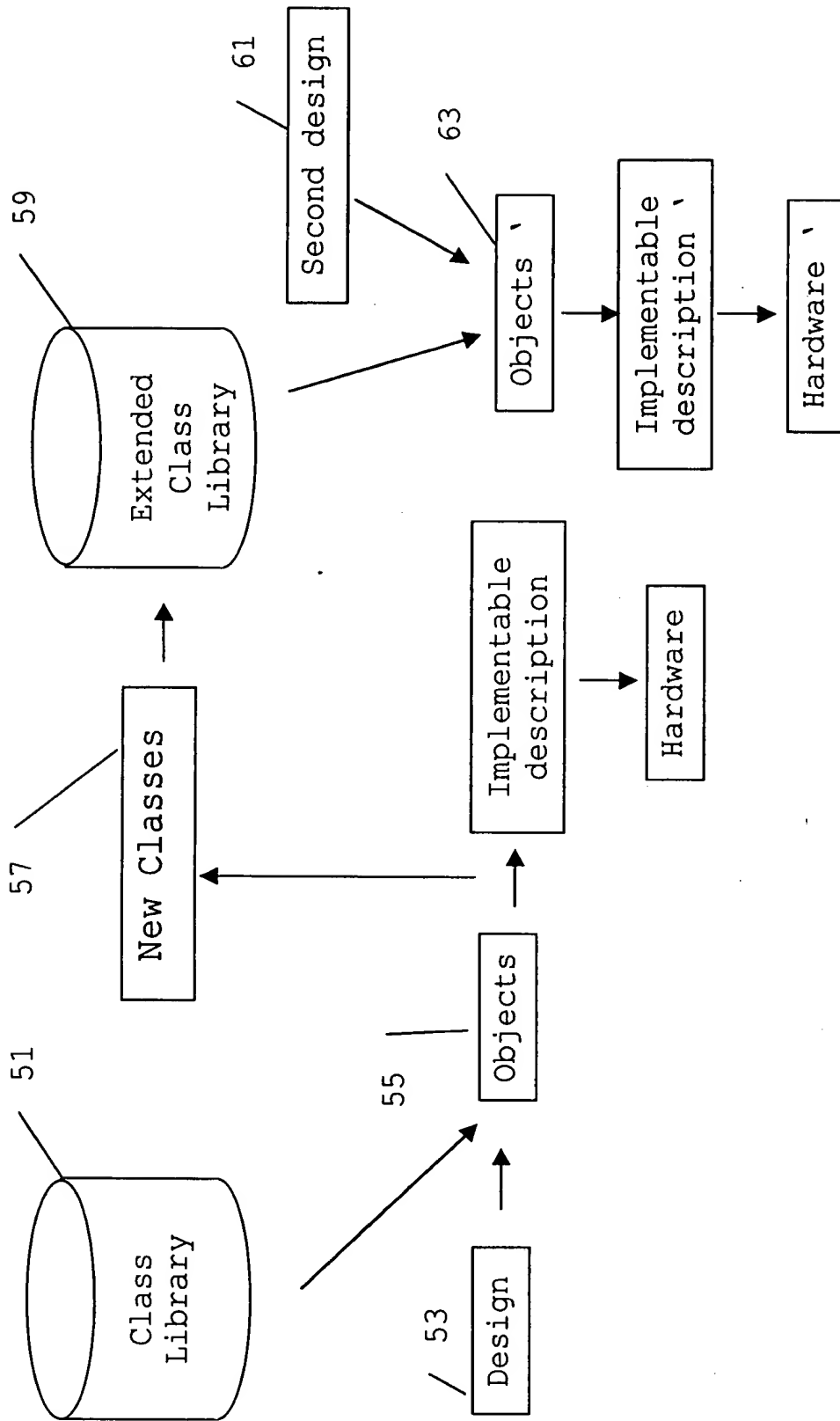


Figure 8